NY PROPERTY DATA
FRAUD ANALYSIS REPORT

DSO 562 – FRAUD ANALYTICS

GROUP 2
Bo Kyung Cho, Rajat Gaur, Puhsin Huang, Yi-Hsin Chung, Swapna Kutcharlapati, John Kim, and Yura Shakhnazaryan

Table of Contents

Executive Summary

This report examines the 2010-2011 New York property dataset, which contains over a million property records in New York City and highlights abnormalities and potential fraud events in the dataset. To solve this project, our team focused on first understanding the nature of real estate fraud inherent in this dataset. According to US Legal, Inc, real estate fraud takes many forms, including common schemes such as foreclosure bailout, home equity fraud, mortgage fraud, and deceptive timeshare scams.

We then explored the dataset through a Data Quality Report (DQR), optimized the overall modeling structure, and gleaned insights from the results of the analysis. We adopted analysis methods including Principal Component Analysis (PCA), Heuristic Algorithm, and Autoencoder, and conducted all data analysis and manipulation using Python and Jupyter Notebook.

Major components of the analysis include:

1. Data cleaning and filling in missing fields
2. Designing and building new variables
3. Dimensionality reduction through PCA process
4. Calculating and comparing fraud score using heuristic algorithm and autoencoder methods

We selected a total of ten unique, abnormal records with highest fraud scores, using both Heuristic Algorithm and Autoencoder, and classified them as potential fraud occurrences. Through further investigation, we explained why these ten records have extremely high fraud scores.

Our explanations for these latent fraud records include:

1. Incorrect data inputs
2. Mortgage fraud
3. Tax avoidance

Please note that, due to the nature of and time dedicated to the analysis, our report may have the following limitations:

1. Limited understanding of New York property legislation
2. Lack of thorough understanding of certain variables
3. Partial optimization of parameters from the unsupervised learning model

Data Description

Overview from NYC Government

"The Offices of the City Register maintain the New York City public records for the Bronx, Brooklyn, Manhattan, and Queens. These records include Real and Personal Property transfers, interest, and ownership information. The Richmond County Clerk maintains all property records for Staten Island. These records are open for inspection and must be recorded and corrected through the office of the Richmond Records dated after 1966 can be recorded and corrected through the Automated City Register Information online or at ACRIS terminals in City Register Offices. Records dated before 1966 can only be accessed City Register's Office in the borough where the property is located."

This New York property dataset represents the annual property valuation and assessment provided by Department of Finance (DOF). The data is collected and entered in the system by various City employees, such as Property Assessors, Property Exemption specialists, ACRIS reporting, Department of Building reporting, and so on. The dataset has 32 fields and 1,070,994 records, and all records are within year 2010 and 2011.

File Name: Property_Valuation_and_Assessment_Data.xlsx
URL: https://data.cityofnewyork.us/Housing-Development/Property-Valuation-and-Assessment-Data/rgy2-tti8
Data Provided by: Department of Finance (DOF)
Dataset Owner: NYC OpenData
Category: Housing & Development
Date Created: September 2, 2011
Last Updated: September 10, 2018
Data Volume: 1,070,994
Fields: 32 fields
Field Variables: RECORD, BBLE, B, BLOCK, LOT, EASEMENT, OWNER, BLDGCL, TAXCLASS, LTFRONT, LTDEPTH, EXT, STORIES, FULLVAL, AVLAND, AVTOT, EXLAND, EXTOT, EXCD1, STADDR, ZIP, EXMPTCL, BLDFRONT, BLDDEPTH, AVLAND2, AVTOT2, EXLAND2, EXTOT2, EXCD2, PERIOD, YEAR, and VALTYPE.

Table 1: Field Names with Description

Field Name	Type	Description
RECORD	*Categorical*	Record number
BBLE	*Categorical*	Concatenation of BORO, BLOCK, LOT, EASEMENT
B	*Categorical*	BORO or Borough
BLOCK	*Categorical*	Valid block ranges by BORO Codes
LOT	*Categorical*	Unique # of the property within BORO/BLOCK
EASEMENT	*Categorical*	Used to describe easement
OWNER	*Categorical*	Owner's name
BLDGCL	*Categorical*	Building class
TAXCLASS	*Categorical*	Tax class
LTFRONT	*Numerical*	Lot frontage in feet
LTDEPTH	*Numerical*	Lot depth in feet
EXT	*Categorical*	No info
STORIES	*Numerical*	Number of stories for the building
FULLVAL	*Numerical*	Total market value of the property
AVLAND	*Numerical*	Assessed land value of the property
AVTOT	*Numerical*	Assessed total value of the property
EXLAND	*Numerical*	Part of land value that is tax exempted
EXTOT	*Numerical*	Part of total assessed value that is tax exempted
EXCD1	*Categorical*	No info
STADDR	*Categorical*	Street address
ZIP	*Categorical*	Postal zip code of the property
EXMPTCL	*Categorical*	Exempt class used for fully exempt properties
BLDFRONT	*Numerical*	Building frontage in feet
BLDDEPTH	*Numerical*	Building depth in feet
AVLAND2	*Numerical*	No info
AVTOT2	*Numerical*	No info
EXLAND2	*Numerical*	No info
EXTOT2	*Numerical*	No info
EXCD2	*Categorical*	No info
PERIOD	*Categorical*	No info
YEAR	*Date/ Time*	Assessment year
VALTYPE	*Categorical*	No info

Data Cleaning

Imputing missing values in fields

A. Necessary missing fields in the NY data

Since the NY Property dataset is large and relatively messy, we prepared the data for our analysis by identifying the necessary missing fields below:

Fields that capture property value ($, numeric variable)

1. FULLVAL. Current year's Total market value of the property
2. AVLAND. Current year's Assessed land value of the property
3. AVTOT. Current year's Assessed total value of the property

Fields that capture property measurements (feet, numeric variable)

4. LTFRONT. Lot Frontage
5. LTDEPTH. Lot Frontage
6. BLDFRONT. Building Frontage
7. BLDDEPTH. Building Frontage

Fields that capture location/address data

8. ZIP. Postal Zip code of the property. It is a nominal variable, having 197 unique values

Others

9. STORIES. Number of stories for the building; ordered (discrete) numeric variable

B. Logic for filling in the missing variables

We filled the missing fields with the most typical value for that field for that record. As the distributions of these fields is severely right-skewed, we filled in the missing records by median instead of mean. Doing so resulted in a more accurate estimate as the median is immune to the effect of outliers.

1. Fields for **property value** (FULLVAL, AVLAND, AVTOT) and **property measurements** (LTFRONT, LTDEPTH, BLDFRONT, BLDDEPTH)
   - We aggregated by ZIP, then TAXCLASS, and used the median value of that group
   - If there are fewer than 5 records in that group, we aggregated by B (borough) & TAXCLASS
   - If there are fewer than 5 records in the above group, we aggregated by B (borough) only

- We considered neighborhood (ZIP code) to be a relatively good predictor since lot area for properties would be comparable neighborhood-to-neighborhood. Similarly, TAXCLASS group similar buildings by incorporating building class, no. of units, no. of apartments, utilities, etc.
- Therefore, we hypothesized that a combination of ZIP and TAXCLASS shows similar buildings in a neighborhood and considered the median of those values

(*Notes*:
[1] **LOT** is not used here, as it has high cardinality (6,366 unique values), and is thus impractical as a field to group on, and thus unusable to fill in values)

[2] **BLDGCL** is not used here, as the 'Property_Valuation_and_Assessment_Data info.xlsx' file states that "there is a direct correlation between the Building Class and the Tax Class")

2. Fields for **ZIP**
   - First, we used address in STADDR field to fill in the ZIP code. If there are several values in one group, we used the mode of that group.
   - If values are not available or needed confirmation, we aggregated the records by B (borough) and BLOCK and used the matching zip code. If there are several values in one group, we used the mode of that group. For example, for 'B1' and 'BLOCK 15' below, we have two zip codes available. Therefore, we used 10280 since it has the highest number of records.

B	BLOCK	ZIP	RECORD
1	1	10004.0	2
	2	10004.0	2
	3	10004.0	4
	4	10004.0	53
	5	10004.0	9
	6	10004.0	2
	7	10004.0	15
	8	10004.0	6

B	BLOCK	ZIP	RECORD
	9	10004.0	4
	10	10004.0	8
	11	10004.0	16
	12	10004.0	2
	13	10004.0	4
	15	10004.0	5
		10280.0	285

Figure 1: Filling in missing Zip Fields

(*Note:* **Borough, Block, Lot and Easement** is the parcel number system used to identify each unit of real estate in New York City. It is an 11-character combination of 1-digit borough, up to 5-digits' block number; up to 4-digits' lot number, 2-digit easement code)

3. Fields for **STORIES**
   - We used median of all values of properties in a ZIP code, TAXCLASS and Building Area, if applicable. Here, Building Area is a calculated field (in square feet), such that Building Area = BLDFRONT x BLDDEPTH
   - We divided Building Area into 10 different bins of equal size and used the median value of that bin to impute missing values in the STORIES field
   - If there are fewer than 5 records in that group, we aggregated records by ZIP code and TAXCLASS only, and used the median value of that group

Variable Creation

To detect abnormalities efficiently, we needed further information beyond the original dataset. So, we created 45 variables according to the following method, after imputing the missing values.

1. Normalize FULLVAL, AVLAND, and AVTOT by Lot area, Building Area and Building Volume to check if they are too small or large

   - Create size variables (Lot area, Building Area and Building Volume) as following:

     Lot Area (lotarea) = LTFRONT * LTDEPTH

     Building Area (bldarea) = BLDFRONT * BLDDEPTH

     Building Volume(bldvol) = bldarea * STORIES

   - Divide FULLVAL, AVLAND, and AVTOT by each size variable:

```
#Make 3 sizes
data['lotarea']=data['LTFRONT']*data['LTDEPTH']
data['bldarea']=data['BLDFRONT']*data['BLDDEPTH']
data['bldvol']=data['bldarea']*data['STORIES']
```

```
#Using above sizes, calculate ratios for FULLVAL, AVLAND, AVTOT variables
data['r1']=data['FULLVAL']/data['lotarea']
data['r2']=data['FULLVAL']/data['bldarea']
data['r3']=data['FULLVAL']/data['bldvol']
data['r4']=data['AVLAND']/data['lotarea']
data['r5']=data['AVLAND']/data['bldarea']
data['r6']=data['AVLAND']/data['bldvol']
data['r7']=data['AVTOT']/data['lotarea']
data['r8']=data['AVTOT']/data['bldarea']
data['r9']=data['AVTOT']/data['bldvol']
```

After this step, we had a total of 9 ratios ranging from r1 to r9.

<Output example>

r1	r2	r3	r4	r5	r6	r7	r8	r9
40.917782	680.142385	13.602848	8.079350	134.296339	2.685927	18.413002	306.064073	6.121281
34675.254965	6159.420290	123.188406	2560.386473	454.805492	9.096110	15603.864734	2771.739130	55.434783
261.796157	261.796157	87.265386	97.551991	97.551991	32.517330	117.808271	117.808271	39.269424
89.714219	836.980890	418.490445	34.913021	325.717946	162.858973	40.371399	376.641401	188.320700
669.074647	53676.325646	53676.325646	297.434763	23861.620343	23861.620343	301.083591	24154.346541	24154.346541

2. Create grouped averages of these 9 variables and group by Zip5, Zip3, TAXCLASS, Borough and All (no grouping)

   - Group by Zip5, Zip3, TAXCLASS, Borough and take average of each of the 9 ratios

```
#group by zip5, zip3, TAXCLASS, borough, all (no group) and calculate average for each group
ratio=['r1','r2','r3','r4','r5','r6','r7','r8','r9']

for i in ratio:
    data[i+'_zip5']=data.groupby('ZIP_new')[i].transform(lambda x:x.mean())
    data[i+'_zip3']=data.groupby('zip3')[i].transform(lambda x:x.mean())
    data[i+'_TAXCLASS']=data.groupby('TAXCLASS')[i].transform(lambda x:x.mean())
    data[i+'_B']=data.groupby('B')[i].transform(lambda x:x.mean())
```

<Output example>

	r1_zip5	r1_zip3	r1_TAXCLASS	r1_B	r2_zip5	r2_zip3	r2_TAXCLASS	r2_B	r3_zip5	r3_zip3	...	r7_TAXCLASS
0	338.47145	381.24211	235.229165	372.217809	434.624165	575.272231	688.056483	563.266278	135.45346	145.051007	...	105.89514
1	338.47145	381.24211	235.229165	372.217809	434.624165	575.272231	688.056483	563.266278	135.45346	145.051007	...	105.89514
2	338.47145	381.24211	235.229165	372.217809	434.624165	575.272231	688.056483	563.266278	135.45346	145.051007	...	105.89514
3	338.47145	381.24211	235.229165	372.217809	434.624165	575.272231	688.056483	563.266278	135.45346	145.051007	...	105.89514
4	338.47145	381.24211	235.229165	372.217809	434.624165	575.272231	688.056483	563.266278	135.45346	145.051007	...	105.89514

3. Divide each of the 9 ratios by grouped average we created above to check if there are any unusual records compared to their peers

```
ratio=['r1','r2','r3','r4','r5','r6','r7','r8','r9']

for i in ratio:
    data[i+'_zip5_'+'ratio']=data[i]/data[i+'_zip5']
    data[i+'_zip3_'+'ratio']=data[i]/data[i+'_zip3']
    data[i+'_TAXCLASS_'+'ratio']=data[i]/data[i+'_TAXCLASS']
    data[i+'_B_'+'ratio']=data[i]/data[i+'_B']
    data[i+'_all_'+'ratio']=data[i]/data[i].mean()
```

<Output example>

```
new_data=data.loc[:,'r1_zip5_ratio':]
new_data.head()
```

	r1_zip5_ratio	r1_zip3_ratio	r1_TAXCLASS_ratio	r1_B_ratio	r1_all_ratio	r2_zip5_ratio	r2_zip3_ratio	r2_TAXCLASS_ratio	r2_B_ratio	r2_all_ratio
0	0.120890	0.107328	0.173949	0.109930	0.192651	1.564898	1.182297	0.988498	1.207497	1.148437
1	102.446617	90.953371	147.410526	93.158506	163.259630	14.171831	10.706966	8.951911	10.935184	10.400332
2	0.773466	0.686693	1.112941	0.703341	1.232601	0.602351	0.455082	0.380486	0.464782	0.442049
3	0.265057	0.235321	0.381391	0.241026	0.422397	1.925758	1.454930	1.216442	1.485942	1.413263
4	1.976754	1.754986	2.844352	1.797535	3.150168	123.500555	93.305956	78.011511	95.294762	90.633792

4. Z-scale the variables using standard z-score formula below.

```
#finalize with z-scaling
new_col=new_data.columns.to_list()
z_scaled=pd.DataFrame()
for i in new_col:
    z_scaled[i]=(new_data[i]-new_data[i].mean())/new_data[i].std()
```

```
z_scaled.head()
```

r1_zip5_ratio	r1_zip3_ratio	r1_TAXCLASS_ratio	r1_B_ratio	r1_all_ratio	r2_zip5_ratio	r2_zip3_ratio	r2_TAXCLASS_ratio	r2_B_ratio	r2_all_ratio	...
-0.236201	-0.135153	-0.143767	-0.133590	-0.158695	0.068317	0.015784	-0.000483	0.016332	0.010989	...
27.256834	13.619209	25.481489	13.831979	31.894268	1.592958	0.840474	0.333776	0.782003	0.695904	...
-0.060866	-0.047436	0.019656	-0.044525	0.045721	-0.048090	-0.047181	-0.026004	-0.042127	-0.041305	...
-0.197466	-0.115775	-0.107664	-0.113914	-0.113536	0.111958	0.039390	0.009085	0.038249	0.030594	...
0.262436	0.114307	0.320994	0.119701	0.422644	14.814811	7.992274	3.232508	7.421985	6.635567	...

Dimensionality Reduction

Many of the expert variables we created could measure correlated or identical properties and are thus redundant. In addition, the sample density decreases when dimensionality increases. So, our next step was to summarize the data with fewer, representative characteristics and avoid overfitting the model through dimensionality reduction. Dimensionality reduction is the process of reducing the number of random variables under consideration by obtaining a set of principal variables. It may be linear or non-linear, depending upon the method used. For this project, we used the prime linear method, called Principal Component Analysis or PCA to remove linear correlations.

PCA is an unsupervised technique used to transform high dimensional data into a smaller dimensional subspace before fitting a machine-learning algorithm. It enables us to summarize the variations in the NY property dataset by lowering dimensions and projecting data onto a new orthogonal-rotated coordinate system.

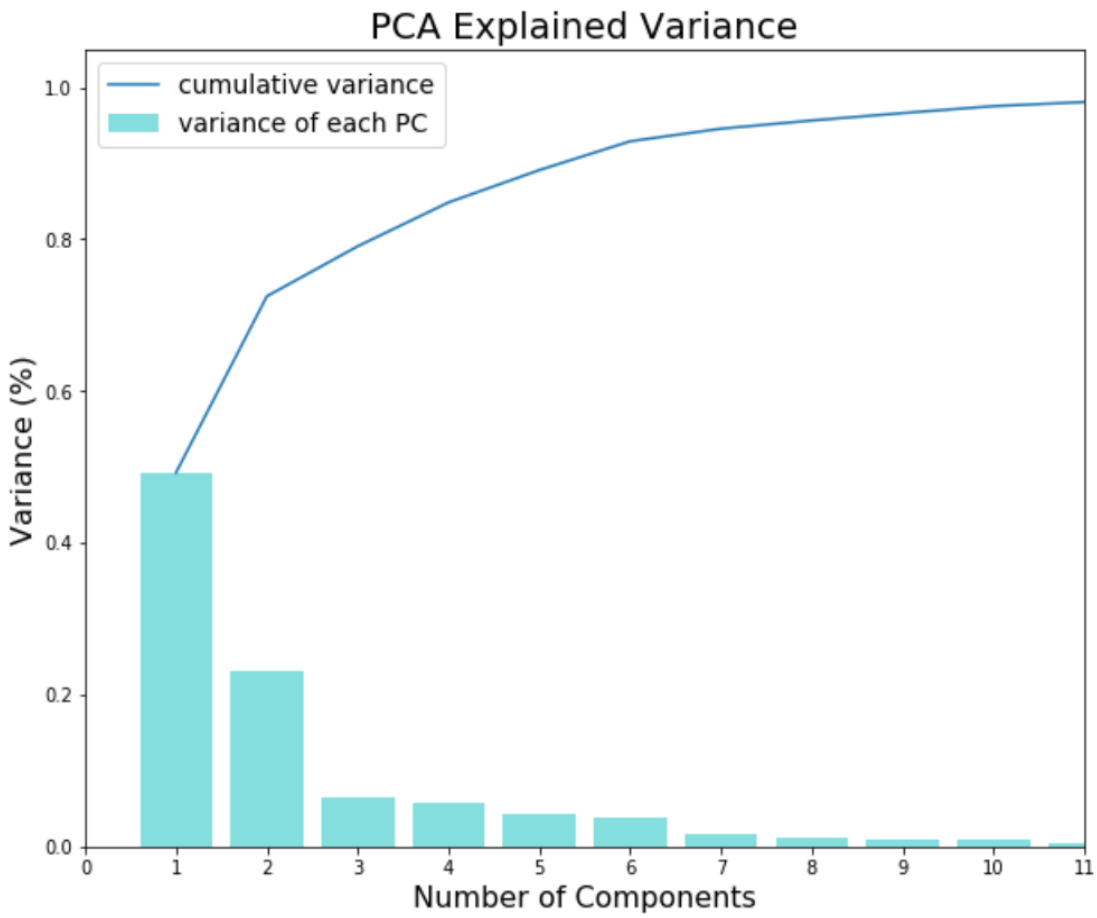


Figure 2: Scree Plot

Before conducting PCA, we Z-scaled the data to provide a fair comparison between the explained variance in the dataset. As Figure 2 shown, the first PC accounted for much of the variation in the dataset with highest Eigenvalue, and the succeeding PCs explains the remaining variation in a decreasing order with decreasing Eigenvalues. We only kept six PCs since they accounted for more than 90% of the total variance. Then, we Z- scaled the data again to make all the retained PCs equally important.

PCA provided insights that would have been hard to obtain from the dataset in its original condition. In addition to dimensionality reduction, PCA helped us capture the variables (i.e., PCs) that explain the most variation of the data with minimum loss of information and without discarding the original fields in the dataset.

The new coordinate system that PCA generates has the origin at the center of the data, so we were able to better visualize and detect potential frauds by calculating the distance between certain records and the origin. The transformed data that we obtained from PCA became crucial for our fraud algorithms, which are respectively Euclidean Distance and Autoencoder. By cutting down variables, we were also able to decrease the training time of algorithms and storage space needed.

Algorithms

Heuristic Algorithm

Before calculating our Fraud Score 1 from Heuristic function, we standardized the values from the results of PCA into Z-scores, and we used these Z-scores to do the rest of our analysis. Implementing Z-scale after PCA helped us calculate the Euclidean distance directly. The Euclidean distance or Euclidean metric is the straight-line distance between two points in Euclidean space. The Euclidean norm, or Euclidean length, or magnitude of a vector measures the length of the vector and takes the following form:

$$\|\mathrm{P}\| = \sqrt{p_1^2 + p_2^2 + .. + p_n^2} = \sqrt{P \times P}$$

Since the new dimensions are relatively uncorrelated and similarly scaled and the data points are centered toward zero, the Z-scores now explicitly indicated the outlierness of each PCA variable. Therefore, we were able to construct the algorithm directly by adding up the Z-scores using the Euclidean distance formula below and calculating anomaly scores for each record.

$$anomalyscore_i = \sqrt{(\sum_k zscore_{ik}^2)}, \text{ where i refers to each record}$$

As expected, the distribution of the anomaly scores based on this heuristic algorithm are extremely right skewed with a long tail. This confirmed that the number of outliers should be small in the dataset.

Autoencoder

Next, we employed a traditional artificial neural network, *autoencoder,* to compile alternative fraud scores and label anomaly records more precisely. We believe that the autoencoder is a relevant algorithm because it helps identify unusual properties by generalizing the initial data and seeking observations that stand out and do not follow universal patterns. Such generalization can be performed by compressing the original records and then recreating them from less available information. Since the learning pattern is primarily shaped by common, abundantly present observations, outlier records would therefore not be expected to be reproduced as well as others.

Since the autoencoder is best utilized for analogous reconstructions, we used it as an additional anomaly evaluation tool. The specifications of its implementation are outlined below:

- The autoencoder was assembled and fitted to train on the previously retrieved normalized data with 1,070,994 records and 6 principal components as its fields

- The computation was performed entirely in Python 3.6, while the implementation of the model was done using TensorFlow.keras library

- We considered the PC data as an input instead of the 45 expert variables because it contained fewer dimensions and already accounted for correlations.

- Similarly, we avoided overcomplicating the reconstruction by keeping the number of weighing parameters minimal. As seen below, this resulted in a relatively simple architecture of the network with dense layers.

```
Model: "model_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_3 (InputLayer)         (None, 6)                 0
_________________________________________________________________
dense_5 (Dense)              (None, 3)                 21
_________________________________________________________________
dense_6 (Dense)              (None, 6)                 24
=================================================================
Total params: 45
Trainable params: 45
Non-trainable params: 0
_________________________________________________________________
```

Figure 3: Autoencoder Model

As this summary table suggests, first layer type (input_3) indicated an input layer with six neurons represented by six principal components placed into the network. The parameters are then generated upon entrance into second layer type (dense_5), which represents the sole hidden layer of the network that acts on three neurons. Finally, the third layer type (dense_6) denoted an output layer, through which the compressed parameters from the previous layer transform into reconstructed values of the initial associated field length, six.

From an optimization perspective, an adaptive learning rate optimization algorithm, *Adam*, which applies stochastic gradient descent in finding 'best-fit' parameters, was compiled. We prioritized *Adam* because it has been empirically proven to significantly ramp up the derivation of an optimal gradient point and the convergence of layer weights to that point. For similar reasons, we used a rectified linear unit (*ReLU*) in the capacity of an activation function of the hidden layer. To execute the network, we used the entire dataset to fit the model, in 100 epochs of 4096 batches of samples each. The figure below shows the pattern of observed loss for that network, measured by mean square error (*MSE)*. As is evident from the figure, the model begins to overfit in the middle stages of processing.

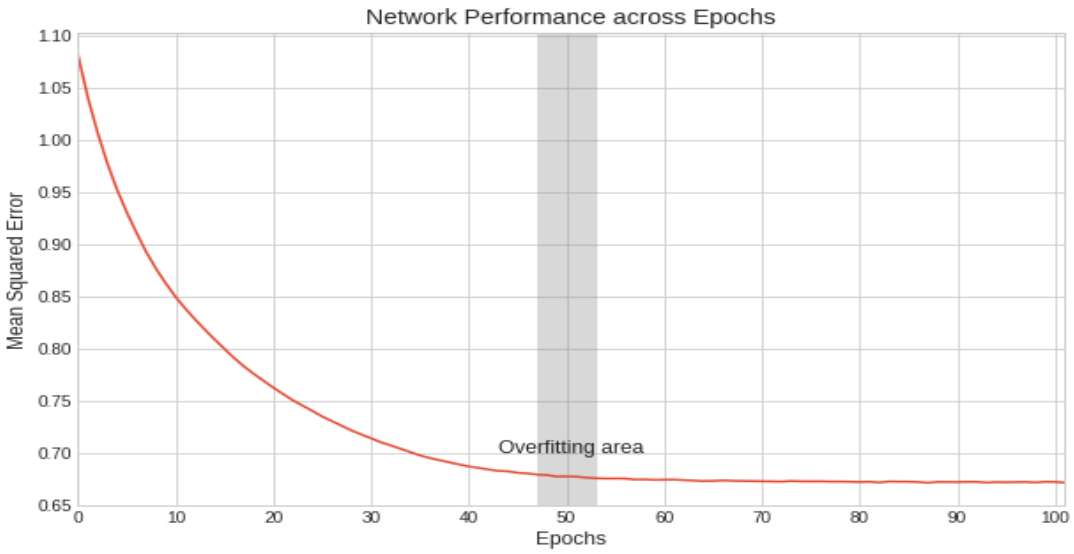


Figure 4: Model loss (y-axis) vs. number of epochs (x-axis) processed

To account for this issue, we called back the respective weights once the effect of overfitting was triggered at the 50th epoch and isolated them as a ground on which the records would be constructed. We then sorted the resulting predicted values, in a form of array with same dimensions as those of the input data, in a data frame, to calculate the reconstruction error.

Scores

To keep the comparison and, later, aggregation of PCA- and autoencoder-generated scores consistent across the board, we applied an L2-norm among every field for every record. However, unlike a Mahalanobis-like, field-wise squared sum that was utilized for the computation of scores for the PCA algorithm, a sum of Euclidean *distances between original and reconstructed values* comprised the scores for the second algorithm.

Once we obtained the Fraud Score 1 (calculated by heuristic algorithm) and Fraud Score 2 (derived from Autoencoder) for each record, we ranked these two columns separately in accordance with the principle of extreme quantile binning, where each unique score was assigned a unique rank. After ranking, each record has two ranking fields which are ranked by Fraud Score 1 and Fraud Score 2 respectively. By creating the two ranking fields, we observed that the autoencoder-related ranking now had the same interpretation as the one created in the course of the heuristic algorithm. Once that criterion met, we combined the distinct rankings into one by taking the average of the two ranking as our Final Fraud Rank, allowing us to better gauge anomalies only in terms of one scale.

The next section of this reports presents the distributions of separate and combined rankings, as well as a deeper look into what makes the most unusual properties so unusual.

Results

The following three graphs show the distribution of Score 1 (heuristic algorithm), Score 2 (Autoencoder), and the Final Rank by combining Score 1 and Score 2. Both Score 1 and Score 2 have extremely right-skewed distribution, meaning that only very few records have unusually high fraud scores.

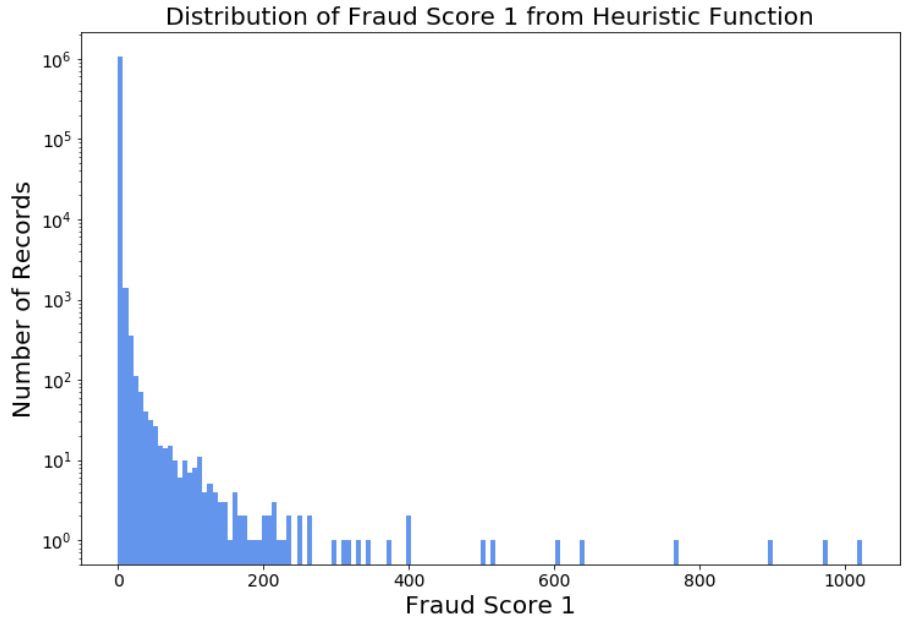


Figure 5: Distribution of Fraud Score from Heuristic Function

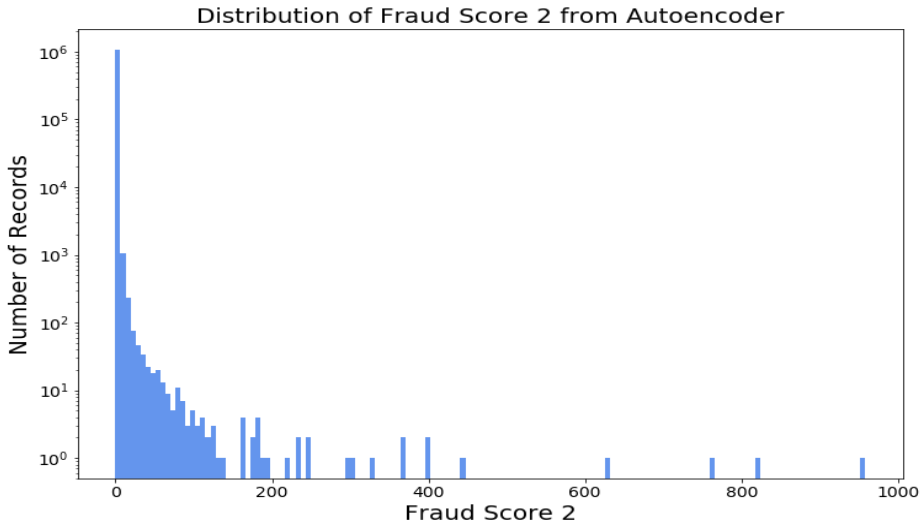


Figure 6: Distribution of Fraud Score from Autoencoder

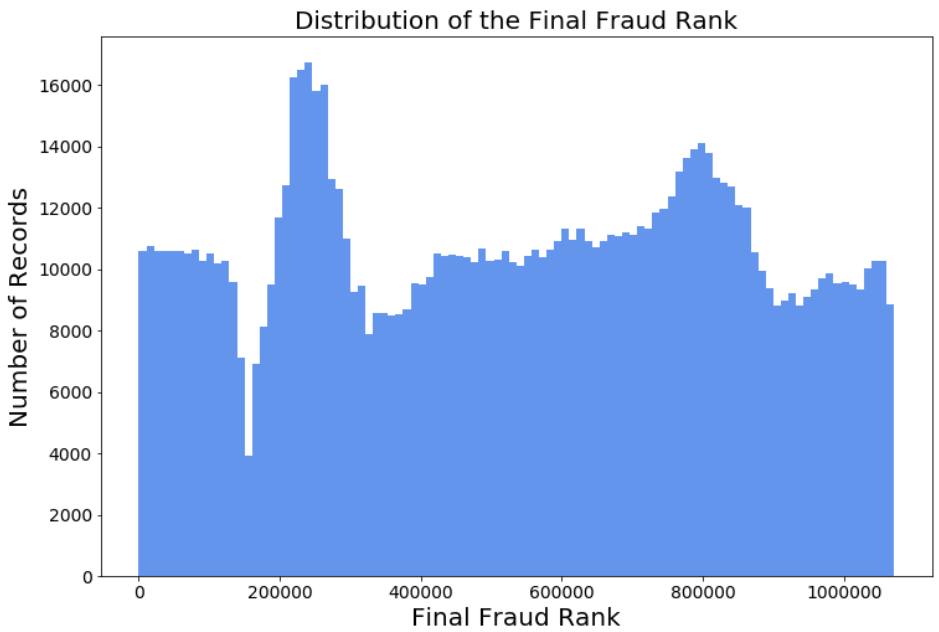


Figure 7: Distribution of Final Fraud Rank

According to the Final Rank, we selected the top 10 records as our top 10 anomalies. Tables 2 and 3 shows these 10 records with 11 original variables in the NY property dataset.

Table 2: Top 10 records of anomalies

RECORD	FULLVAL	AVLAND	AVTOT	LTFRONT	LTDEPTH
632816	2,930,000	1,318,500	1,318,500	157	95
565392	4,326,303,700	1,946,836,665	1,946,836,665	117	108
1067360	836,000	28,800	50,160	1	1
917942	374,019,883	1,792,808,947	4,668,308,947	4,910	100
556609	136,000,000	60,750,000	61,200,000	35	50
85886	70,214,000	31,455,000	31,596,300	4,000	150
585439	3,712,000	252,000	1,670,400	94	165
750816	256,435	6,501	6,501	1	1
185809	53,030	23,864	23,864	1	3
935158	1,040,000	236,250	468,000	136	132

Table 3: Top 10 records of anomalies – Normalized

RECORD	BLDFRONT	BLDDEPTH	ZIP	STORIES	TAXCLASS	B
632816	1	1	11373	1	2	4
565392	42	85	11225	2	4	3
1067360	36	45	10307	2	1	5
917942	40	60	11422	3	4	4
556609	88	62	11236	1	4	3
85886	8	8	10023	1	4	1
585439	1	1	11101	10	4	4
750816	21	40	11367	2	1B	4
185809	55	75	10462	1	4	2
935158	1	1	10301	8	2	5

The descriptions below examine why these 10 records have extremely high fraud scores.

- **RECORD: 632816**

  We noticed that r2, r3, r5, r6, r8, and r9 of this record are all extremely high because its BLDFRONT, BLDDEPTH, and STORIES all have value of 1. However, by searching the building online using its address, we found that the property is a 6-story premier full-service rental building called “The Elm East” at Elmhurst, Queens. It features 83 luxury rental apartments comprising of studios, one-bedrooms, and two-bedrooms. Therefore, the value 1 of the three fields mentioned above must be false entries.

- **RECORD: 565392**

  For this record, FULLVAL, AVLAND, and AVTOT all have values over one billion, making its r1 to r9 all having extremely large values. The information we found that might be helpful to explain its high property values is that this property is owned by the US government and is located on FLATBUSH AVENUE.

- **RECORD: 1067360**

  The LTFRONT and LTDEPTH of this property have values of 1. Thus, they make its r1, r4, and r7 extremely large, explaining why this record is an anomaly. However, from the information found online, the property is "normal". It is considered a multi-family (2-4 units, 2 stories) home.

- **RECORD: 917942**

  After data cleaning, the mean and median of AVTOT are 228,182 and 25,574 respectively. However, The AVTOT of this record is over 4 billion dollars. Clearly, the Assessed Value of this record is an outlier. Similarly, the values of AVLAND and LTFRONT of this record are all extremely large, which leads to extremely high values for r2 to r8. In fact, r5, r8, and r9 all exhibit Z-values of 300-400, indicating that they are 300 to 400 standard deviations further from the mean.

- **RECORD: 556609**

  This record is very unusual in terms of all r1 to r9. When filtered by ZIP, TAXCLASS, STORIES, and B to compare this record with similar properties, all the ratios r1-r9 represent the maximum value in each of the distribution. Therefore, it is obvious that this record is an outlier when we compare it to the average values of r1-r9.

- **RECORD: 85886**

  This record's r2, r3, r5, r6, r8, and r9 are high because its FULLVAL, AVLAND, and AVTOT are all extremely high. At the same time, its BLDFRONT and BLDDEPTH have small values of 8, which make the 6 ratios mentioned above unusually large. However, r1, r4 and r7 are reasonable because LTFRONT of the record is very large as well. As the place is owned by 'PARKS AND RECREATION,' it seems that the property refers to a park area with a small size building.

- **RECORD: 585439**

  For this record, r2, r3, r5, r6, r8, and r9 are all extremely high because BLDFRONT and BLDDEPTH have value of 1. Using the property's address, we found that the property is a 12-floor hotel called "Z NYC Hotel" in Long Island City. Therefore, it is not possible for BLDFRONT and BLDDEPTH to have values of 1. There must have been some mistakes when collecting the data.

- **RECORD: 750816**

  Before filling in missing values, FULLVAL, AVLAND, AVTOT, BLDFRONT, and BLDDEPTH are all missing for this record. Also, both LTFRONT and LTDEPTH equal to 1 so that r1, r4, and r7 of the record are all extremely high, leading to a very high fraud score. Other information we know is that the property is in a residential area.

- **RECORD: 185809**

  The record's LTFRONT and LTDEPTH have a value of 1 and 3 respectively. Thus, they caused r1, r4, and r7 to be extremely high and in turn made the record have extremely high fraud scores. Exact address and owner's name are not provided. However, it might be an industrial property such as factory, plant, or manufacturing real estate since it is in an industrial business zone.

- **RECORD: 935158**

  For this record, r2, r3, r5, r6, r8, and r9 are all extremely high since BLDFRONT and BLDDEPTH have a value of 1. However, we found that the property is actually a 40-unit luxury apartment so the real values of BLDFRONT and BLDDEPTH cannot be 1. There must have been some mistakes when gathering the data.

Conclusions

This report documented how we analyzed over one million NY property data with the goal of detecting anomalies. We first cleaned the data by filling in missing values, created 45 variables, and standardized these 45 fields. Afterwards, we reduced the dimensionality of the data by performing PCA. At the end, we chose the top 6 principle components for further analysis. The next step we did was to derive two fraud scores for each record using heuristic algorithm and autoencoder model. It turned out that the top ten records between the two analysis methods are highly overlapped.

After obtaining these two fraud scores, we ranked them separately and created a final rank by taking the average of the two rankings. By sorting the final rank, we selected the top 10 records as our 10 anomalies. We then examined the original fields of the data and provided our explanation for their extremely high fraud scores.

Through further investigation of these records, we also conclude the following causes of abnormalities.

- Mortgage Fraud: False reporting of land/property values to secure higher loans. This type of fraud is a common practice in real estate fraud, where property owners report higher property value to be able to secure higher loans from finance institutions.

- Tax avoidance: In this fraud type, the property owner tries to avoid tax and evade higher taxes by underrating the property or land value. This method can also include illegal tax exemption.

- Incorrect data input: Based on our observations, some data abnormality could be caused by incorrect input or inaccurate evaluation.

It should be noted that our conclusions are based on limited understanding of the NY Property dataset as well as the NY Real Estate Legislation. Due to the nature of and time dedicated to the analysis, our report may also suffer from the lack of a thorough understanding of certain variables and partial optimization of parameters from the unsupervised learning model.

We believe that further in-depth investigation with more field knowledge and comprehensive information can overcome these limitations.

Appendix: Data Quality Report

Data Overview:

This Data Quality Report summarized information of “NY property data.csv”, which contains a large amount of property records in New York City. The data is provided by Department of Finance (DOF) of NYC government and can be obtained from “Property Valuation and Assessment Data” on NYC OpenData. The data file has 32 fields and 1,070,994 records, and all records are within year 2010 and 2011.

The following contents include the two parts:

- Tables for Summary Statistics of Each Field
- Field Description and Graphs

Tables for Summary Statistics of Each Field

Table 4

General Information					
Field Name	**Field Type**	**#Records Populated**	**% Records Populated (%)**	**# Unique Values**	**# Records w/ Value Zero**
RECORD	Categorical	1070994	100.00	1070994	n/a
BBLE	Categorical	1070994	100.00	1070994	n/a
B	Categorical	1070994	100.00	5	n/a
BLOCK	Categorical	1070994	100.00	13984	n/a
LOT	Categorical	1070994	100.00	6366	n/a
EASEMENT	Categorical	4636	0.43	13	n/a
OWNER	Categorical	1039249	97.04	863347	n/a
BLDGCL	Categorical	1070994	100.00	200	n/a
TAXCLASS	Categorical	1070994	100.00	11	n/a
LTFRONT	Numerical	1070994	100.00	1297	169108
LTDEPTH	Numerical	1070994	100.00	1370	170128
EXT	Categorical	354305	33.08	4	n/a
STORIES	Numerical	1014730	94.75	112	0
FULLVAL	Numerical	1070994	100.00	109324	13007
AVLAND	Numerical	1070994	100.00	70921	13009
AVTOT	Numerical	1070994	100.00	112914	13007
EXLAND	Numerical	1070994	100.00	33419	491699
EXTOT	Numerical	1070994	100.00	64255	432572
EXCD1	Categorical	638488	59.62	130	n/a
STADDR	Categorical	1070318	99.94	839281	n/a
ZIP	Categorical	1041104	97.21	197	n/a
EXMPTCL	Categorical	15579	1.45	15	n/a
BLDFRONT	Numerical	1070994	100.00	612	228815
BLDDEPTH	Numerical	1070994	100.00	621	228853
AVLAND2	Numerical	282726	26.40	58592	0
AVTOT2	Numerical	282732	26.40	111361	0
EXLAND2	Numerical	87449	8.17	22196	0
EXTOT2	Numerical	130828	12.22	48349	0
EXCD2	Categorical	92948	8.68	61	n/a
PERIOD	Categorical	1070994	100.00	1	n/a
YEAR	Date/Time	1070994	100.00	1	n/a
VALTYPE	Categorical	1070994	100.00	1	n/a

Table 5

Numerical Fields							
Field Name	**Mean**	**Standard Deviation**	**Min**	**25%**	**50%**	**75%**	**Max**
LTFRONT	36.64	74.03	0.00	19.00	25.00	40.00	9999.00
LTDEPTH	88.86	76.40	0.00	80.00	100.00	100.00	9999.00
STORIES	5.01	8.37	1.00	2.00	2.00	3.00	119.00
FULLVAL	874264.51	11582430.00	0.00	304000.00	447000.00	619000.00	6150000000.00
AVLAND	85067.92	4057260.06	0.00	9180.00	13678.00	19740.00	2668500000.00
AVTOT	227238.17	6877529.31	0.00	18374.00	25340.00	45438.00	4668308947.00
EXLAND	36423.89	3981575.79	0.00	0.00	1620.00	1620.00	2668500000.00
EXTOT	91186.98	6508402.82	0.00	0.00	1620.00	2090.00	4668308947.00
BLDFRONT	23.04	35.58	0.00	15.00	20.00	24.00	7575.00
BLDDEPTH	39.92	42.71	0.00	26.00	39.00	50.00	9393.00
AVLAND2	246235.72	6178962.56	3.00	5705.00	20145.00	62640.00	2371005000.00
AVTOT2	713911.44	11652528.95	3.00	33912.00	79962.50	240551.00	4501180002.00
EXLAND2	351235.68	10802212.67	1.00	2090.00	3048.00	31779.00	2371005000.00
EXTOT2	656768.28	16072510.17	7.00	2870.00	37062.00	106840.75	4501180002.00

Table 6

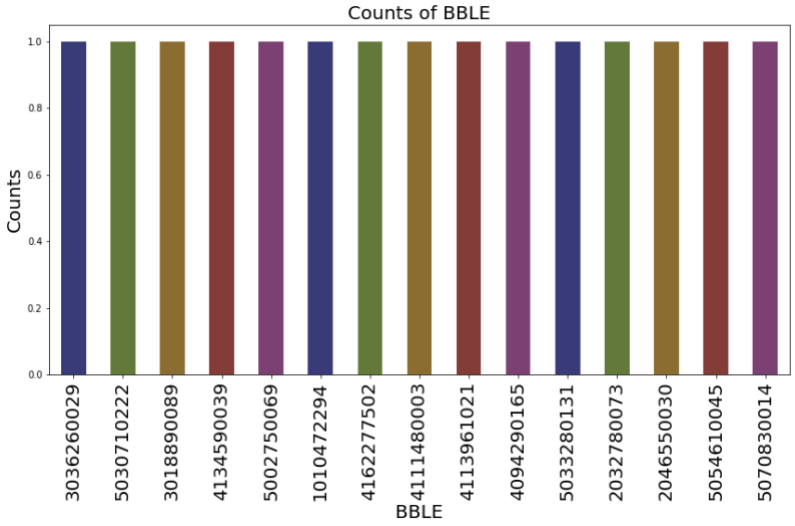
Categorical Fields			
Field Name	**Most Common Value**	**Frequency**	**Percentage (%)**
B	4	358046	33.43
BLOCK	3944	3888	0.36
LOT	1	24367	2.28
EASEMENT	E	4148	0.39
OWNER	PARKCHESTER PRESERVAT	6020	0.56
BLDGCL	R4	139879	13.06
TAXCLASS	1	660721	61.69
EXT	G	266970	24.93
EXCD1	1017	425348	39.72
STADDR	501 SURF AVENUE	902	0.08
ZIP	10314	24606	2.30
EXMPTCL	X1	6912	0.65
EXCD2	1017	65777	6.14
PERIOD	FINAL	1070994	100.00
YEAR	2010/11	1070994	100.00
VALTYPE	AC-TR	1070994	100.00

Field Description and Graphs

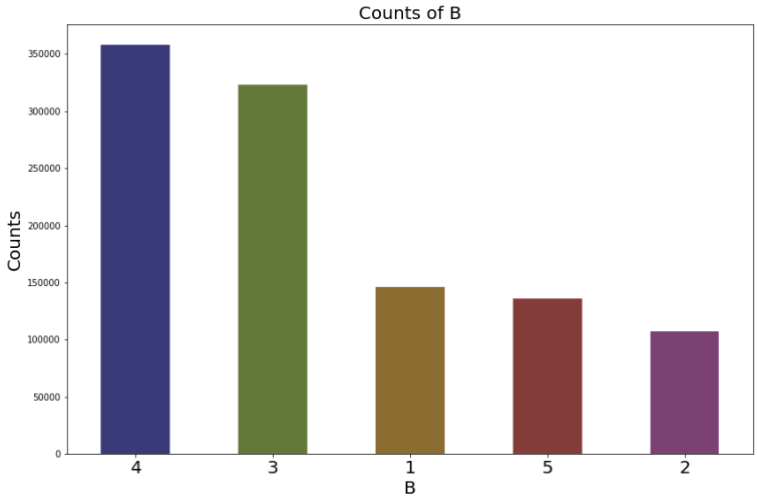
For each numerical field, I showed the distribution of the field with a histogram. For each categorical field, I showed the number of observations for top 15 categories with a bar chart.

1. RECORD: The RECORD field is a unique integer for each observation in the dataset, served for labeling each record.

2. BBLE: The BBLE field is a unique 11-digit alphanumeric text for each field. It is a concatenation of AV_BORO, AV_BLOCK, AV_LOT, and AV_EASEMENT. As you can see in the graph, each value has only one record.

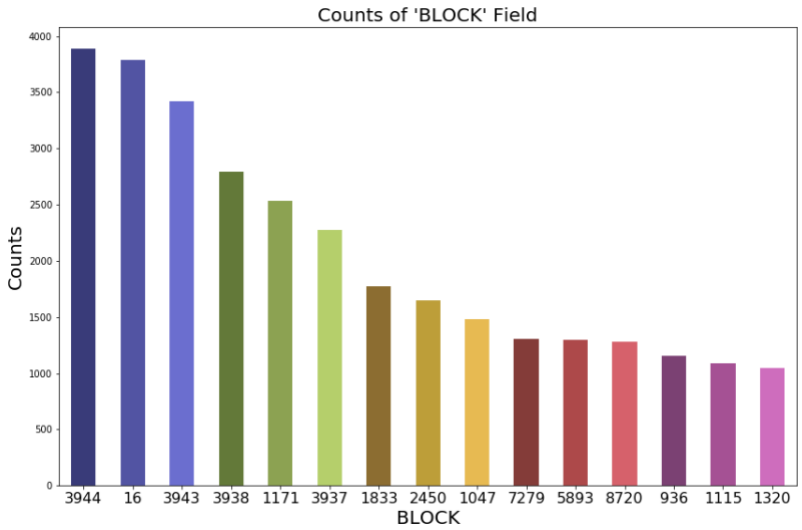


3. B: The B field represents 5 boroughs, and the values for each borough are shown as below.

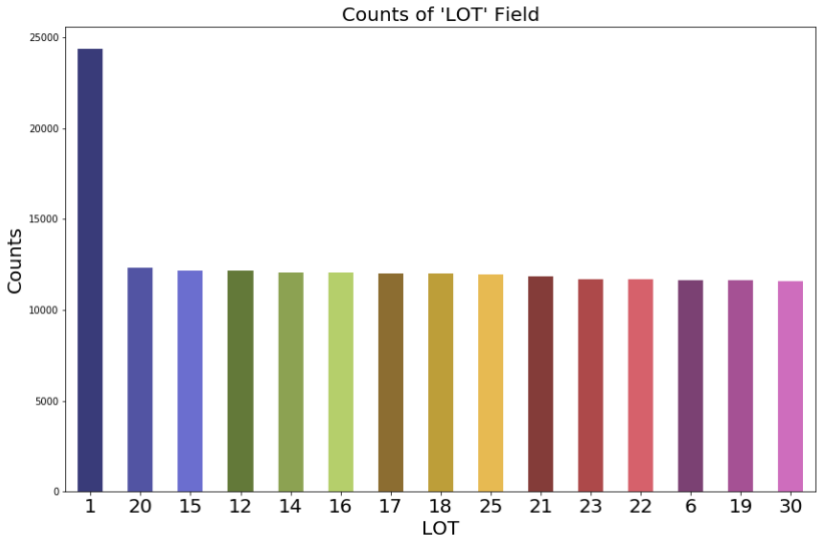


1 = Manhattan. 2 = Bronx. 3 = Brooklyn. 4 = Queens. 5 = Staten Island

4. BLOCK: The BLOCK field represents different block ranges within different boroughs.



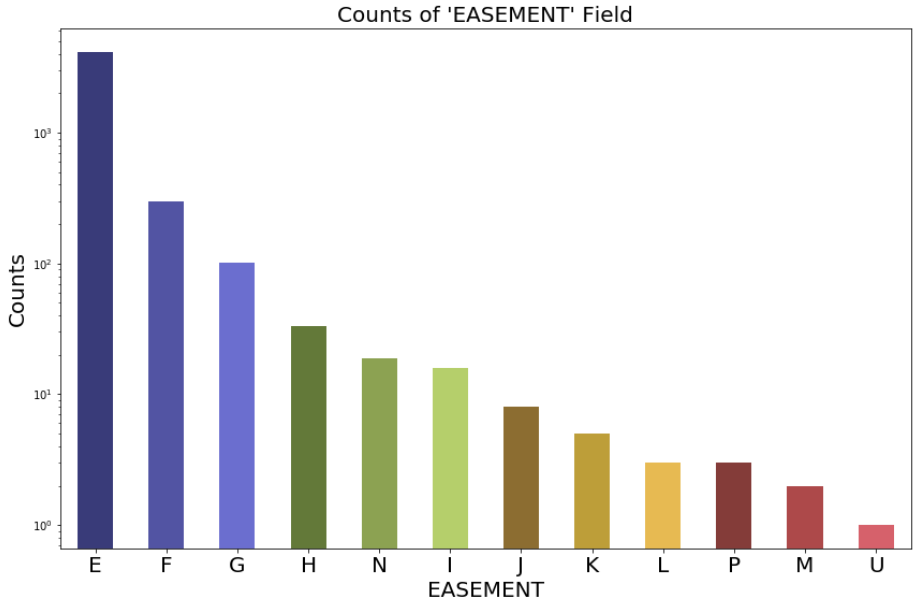
5. LOT: The LOT field is a unique number within each borough or each block for every observation.



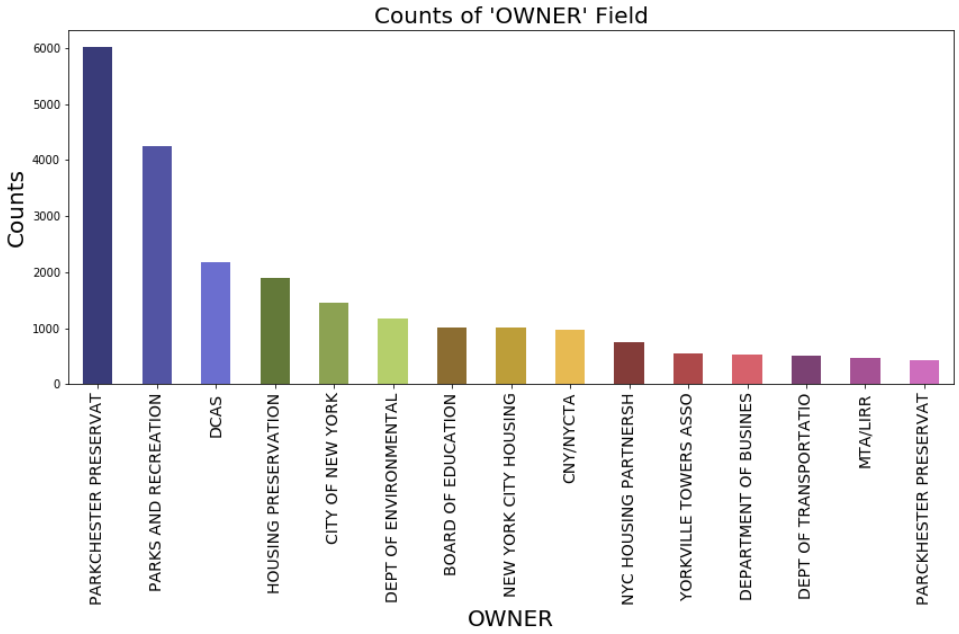
6. EASEMENT: The EASEMENT field is a variable that describes the following.

SPACE = the lot has no Easement
'A' = the portion of the Lot that has an Air Easement
'B' = Non-Air Rights
'E' = the portion of the lot that has a Land Easement
'F' THRU 'M' = Duplicates of 'E'
'N' = Non-Transit Easement
'P' = Piers

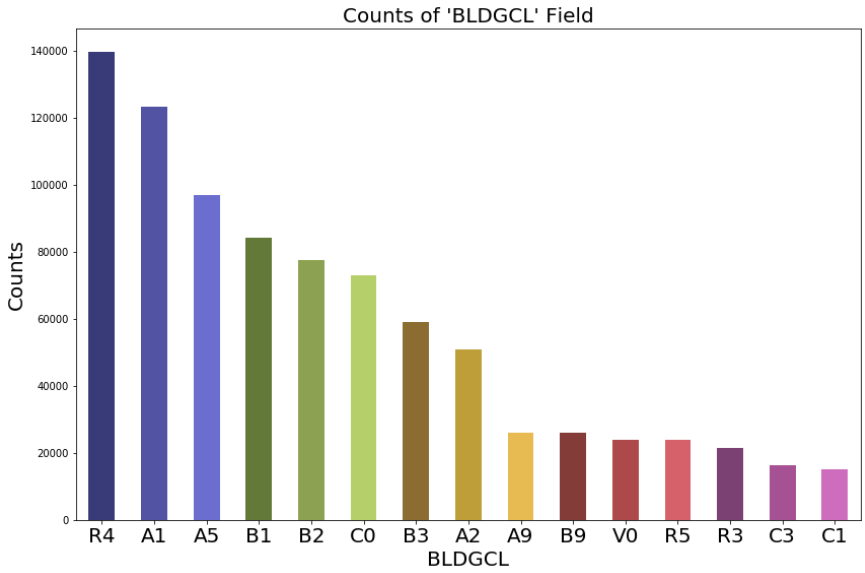
'R' = Railroads
'S' = Street
'U' = U.S. Government



7. OWNER: The OWNER field shows the name of the owner for the property.

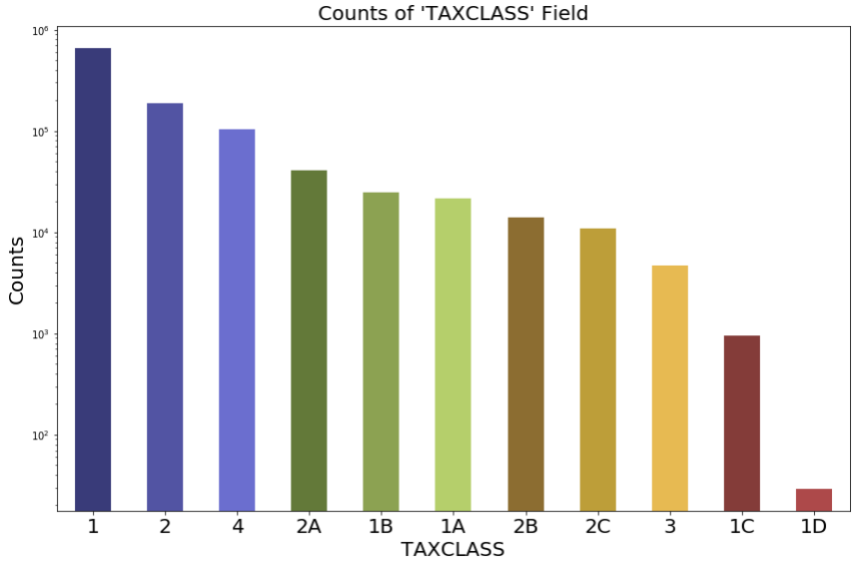


8. BLDGCL: The BLDGCL field indicates the building class and has a direct correlation with TAXCLASS.

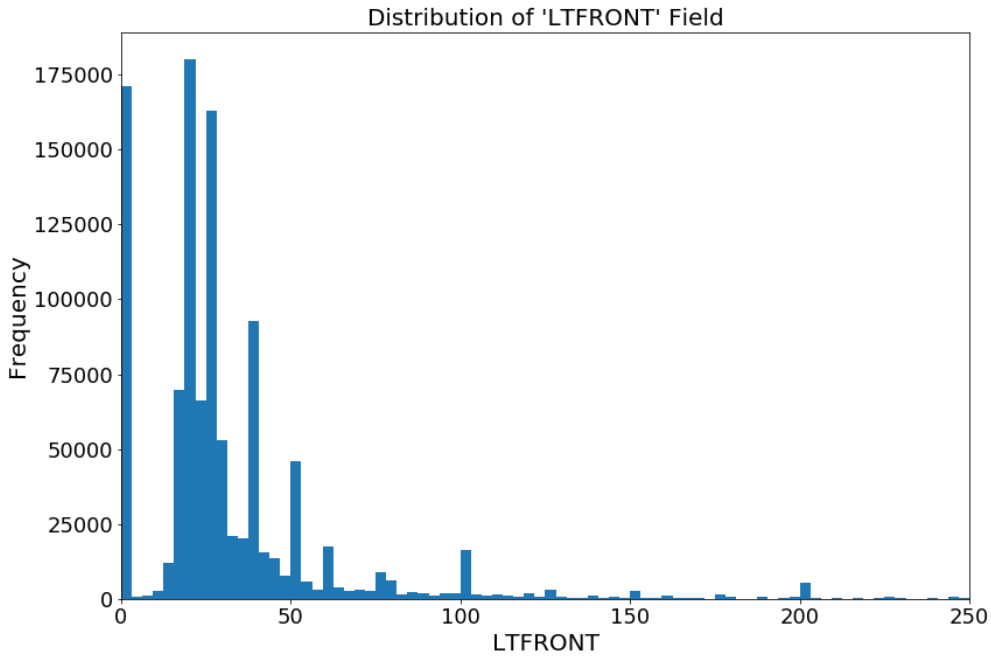


9. TAXCLASS: The TAXCLASS field shows the current property tax class code (NYS Classification), and the first position of the tax class has a direct correlation with the building class.

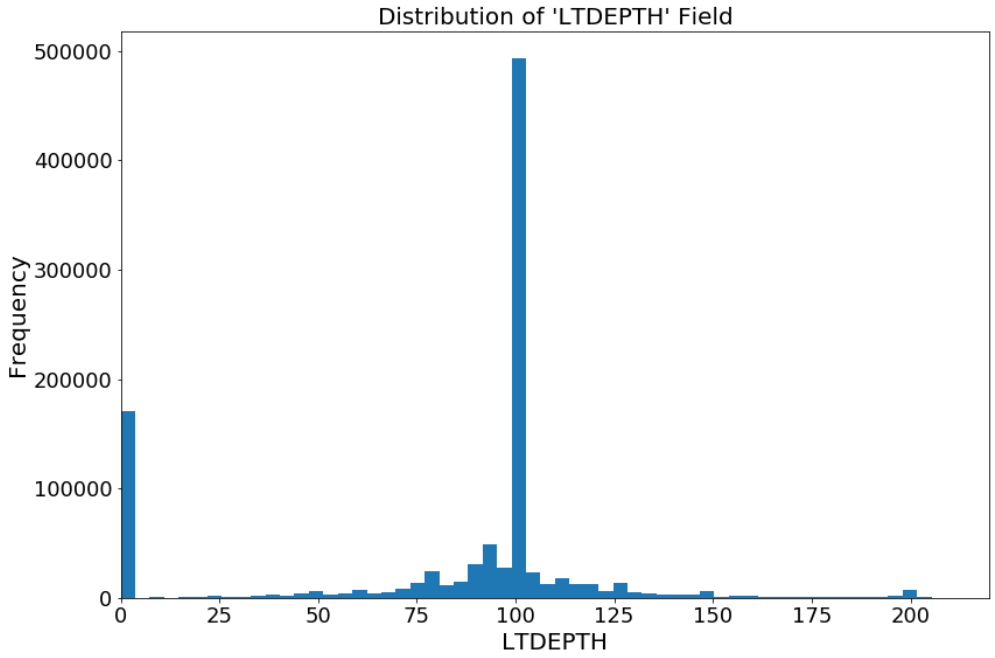
   1 = 1-3 UNIT RESIDENCES
   1A = 1-3 STORY CONDOMINIUMS, ORIGINALLY A CONDO
   1B = RESIDENTIAL VACANT LAND
   1C = 1-3 UNIT CONDOMINUMS, ORIGINALLY TAX CLASS 1
   1D = SELECT BUNGALOW COLONIES
   2 = APARTMENTS
   2A = APARTMENTS WITH 4-6 UNITS
   2B = APARTMENTS WITH 7-10 UNITS
   2C = COOPS/CONDOS WITH 2-10 UNITS
   3 = UTILITIES (EXCEPT CEILING RR)
   4A = UTILITIES - CEILING RAILROADS
   4 = ALL OTHERS



10. LTFRONT: The LTFRONT field is the lot width in feet. It includes records with value <= 250.

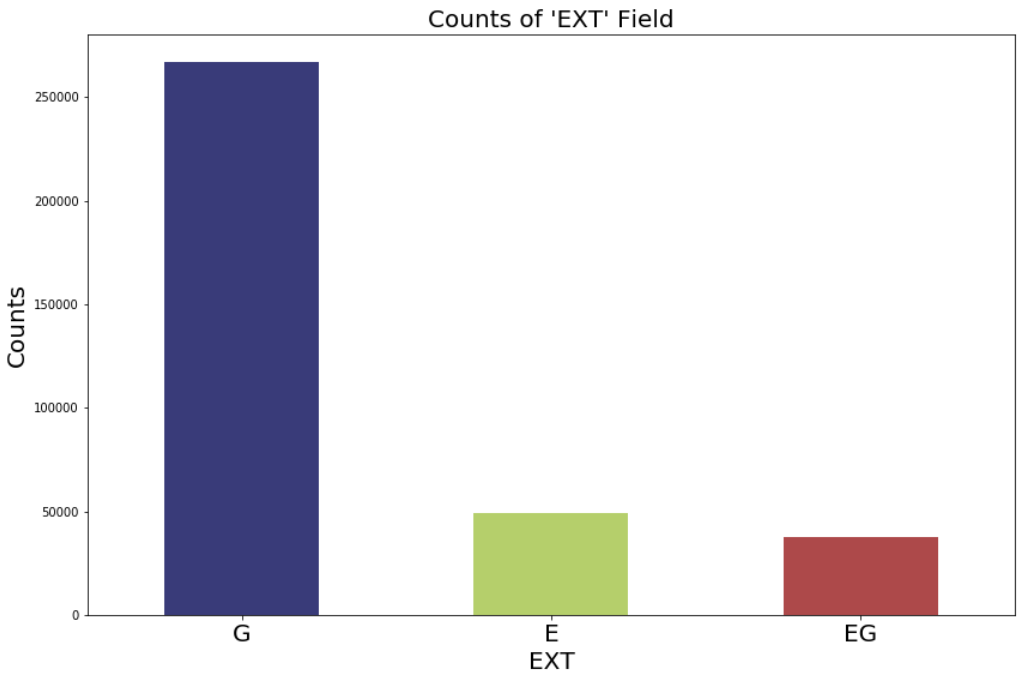


11. LTDEPTH: The LTDEPTH field is the lot depth in feet. It includes records with value <= 220.

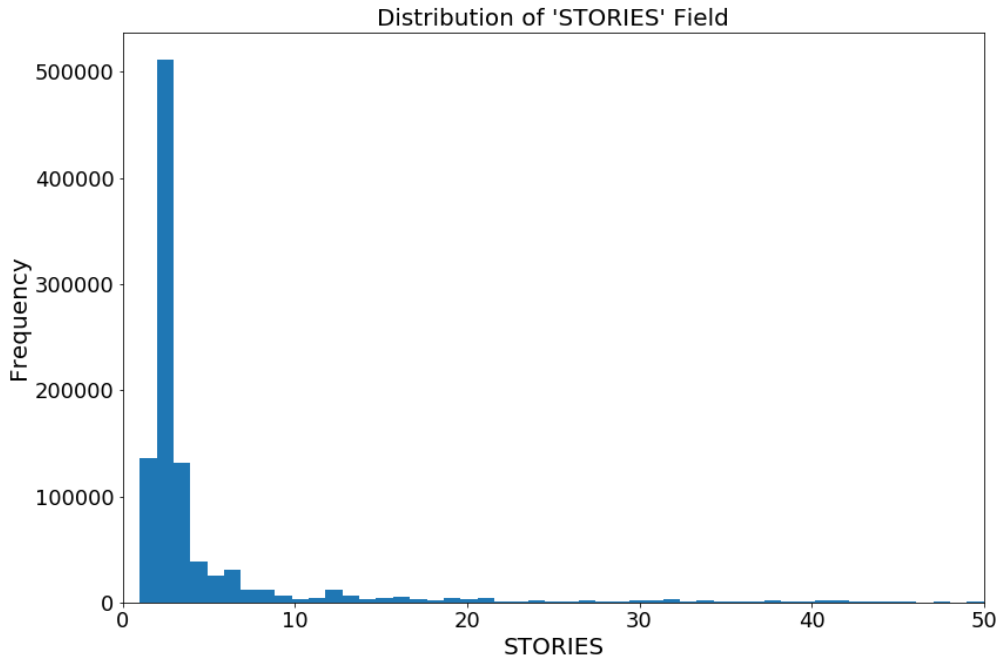


12. EXT: The EXT field indicates the information for extension.

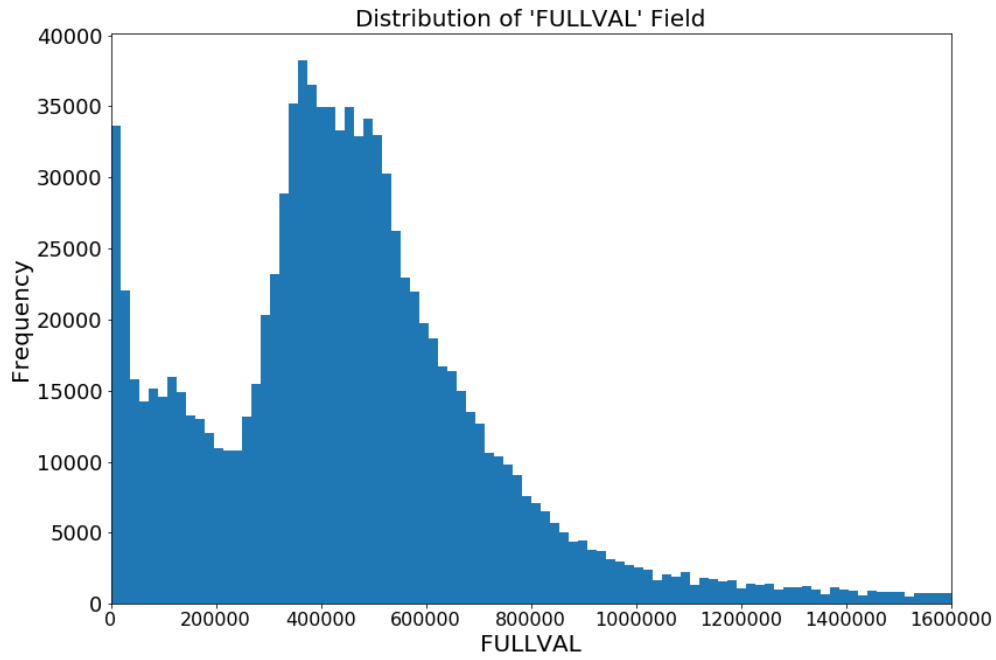
'E' = EXTENSION. 'G' = GARAGE. 'EG' = EXTENSION AND GARAGE.



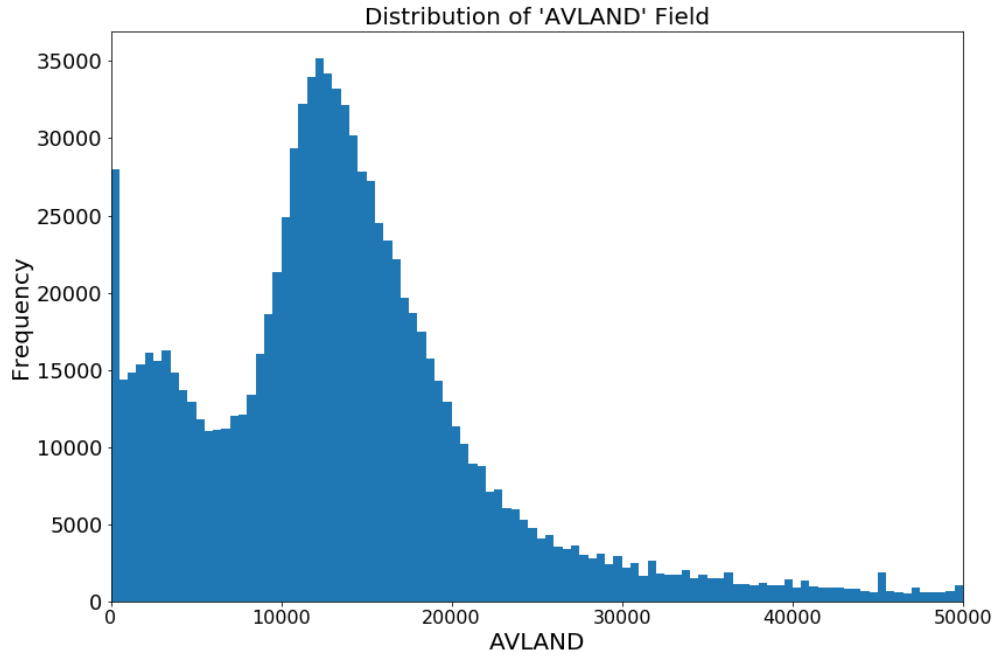
13. STORIES: The STORIES field represents the number of stories for the building (# of Floors). It includes records with value <= 50.



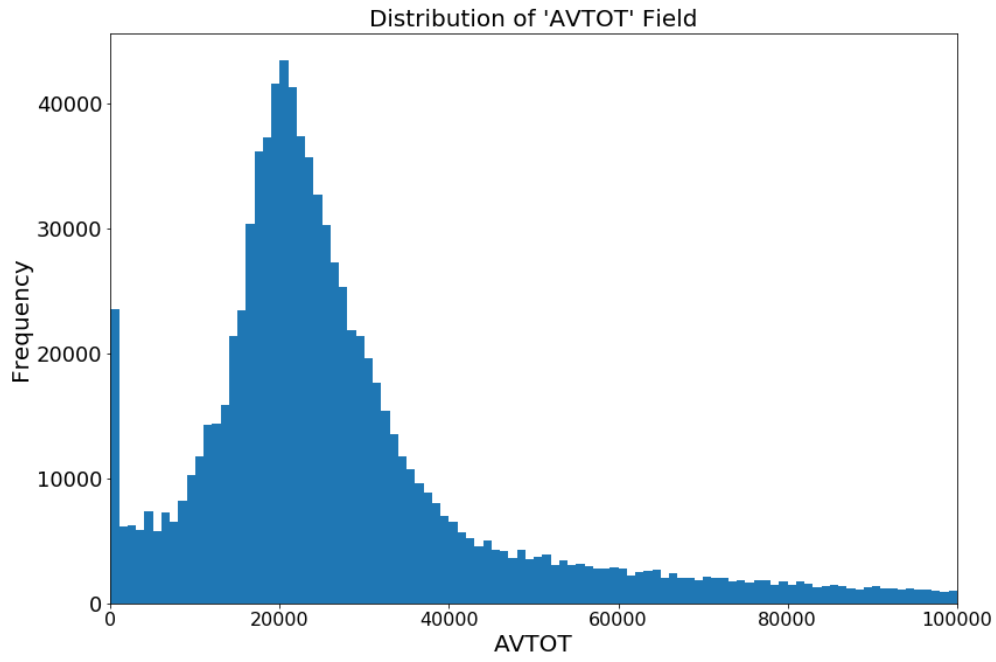
14. FULLVAL: The FULLVAL field is the market value of the property. It includes records with value <= 1600000.



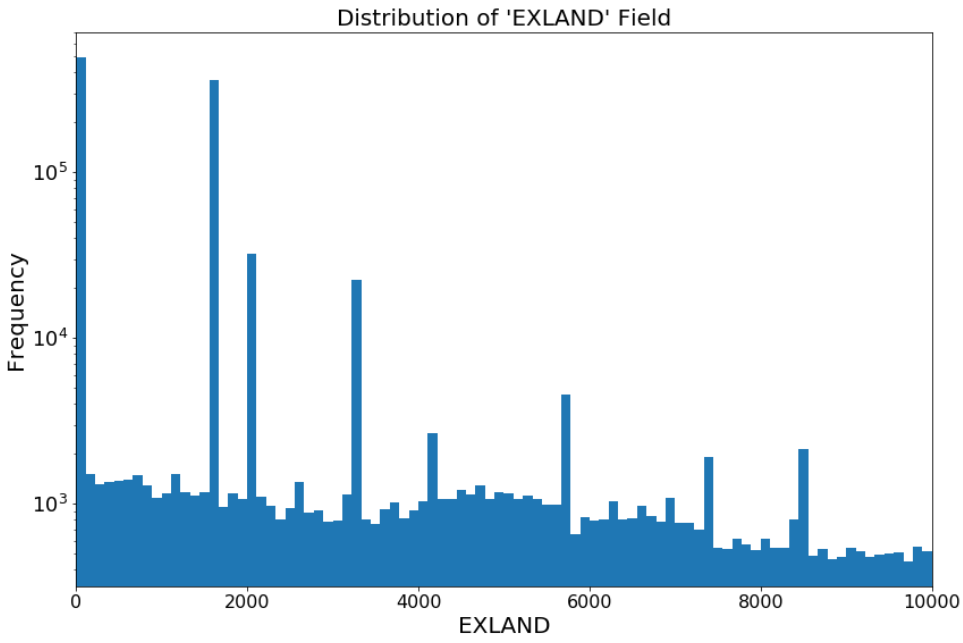
15. AVLAND: The AVLAND field is the actual land value of the property. It includes records with value <= 50000.



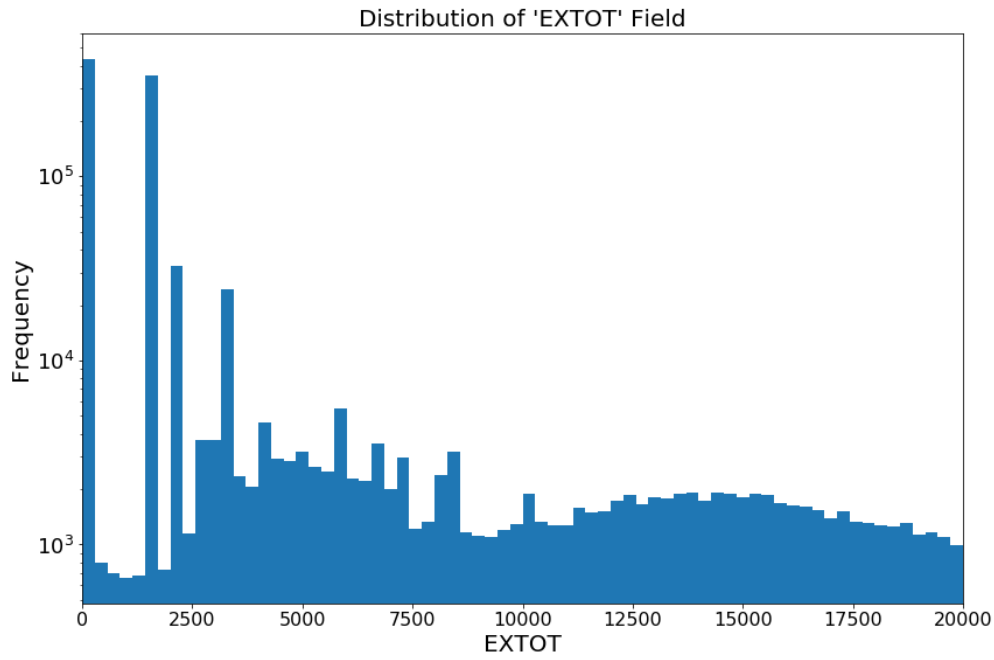
16. AVTOT: The AVTOT field is the actual total value of the property. I include records with value <= 100000.



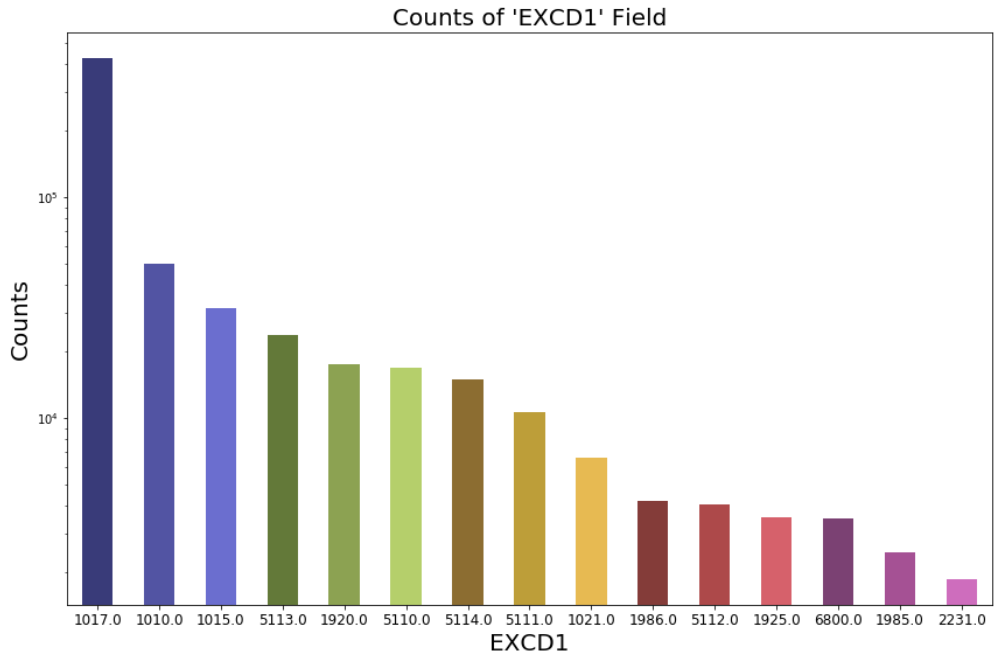
17. EXLAND: The EXLAND field is the actual exempt land value of the property. It includes records with value <= 10000.



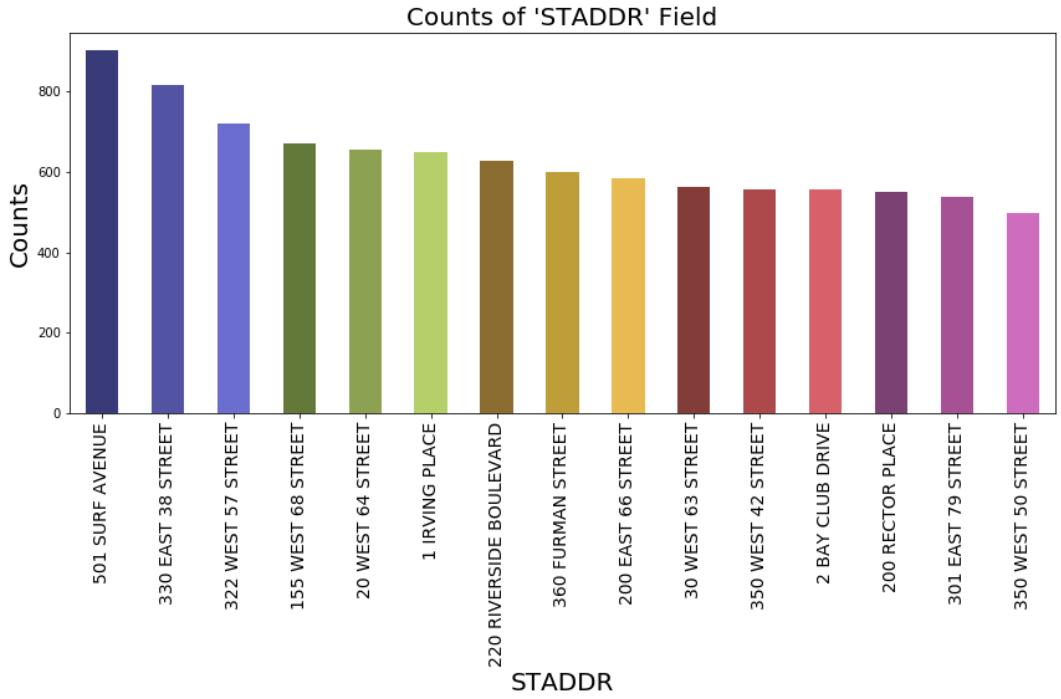
18. EXTOT: The EXTOT field is the actual exempt total value of the property. It includes records with value <= 20000.



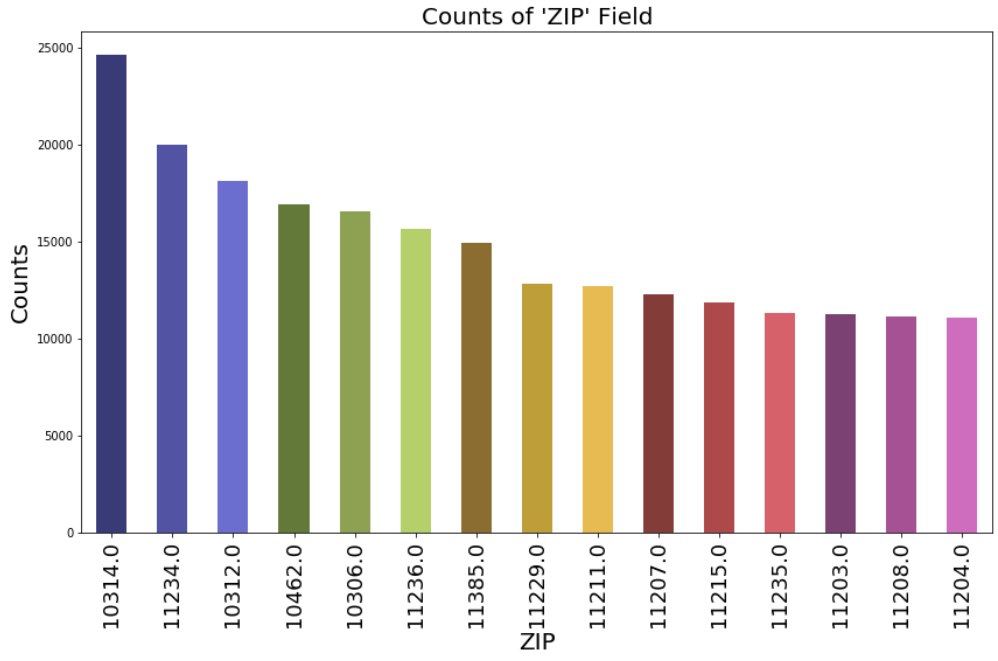
19. EXCD1: The EXCD1 field represents the exemption code 1 for each property.



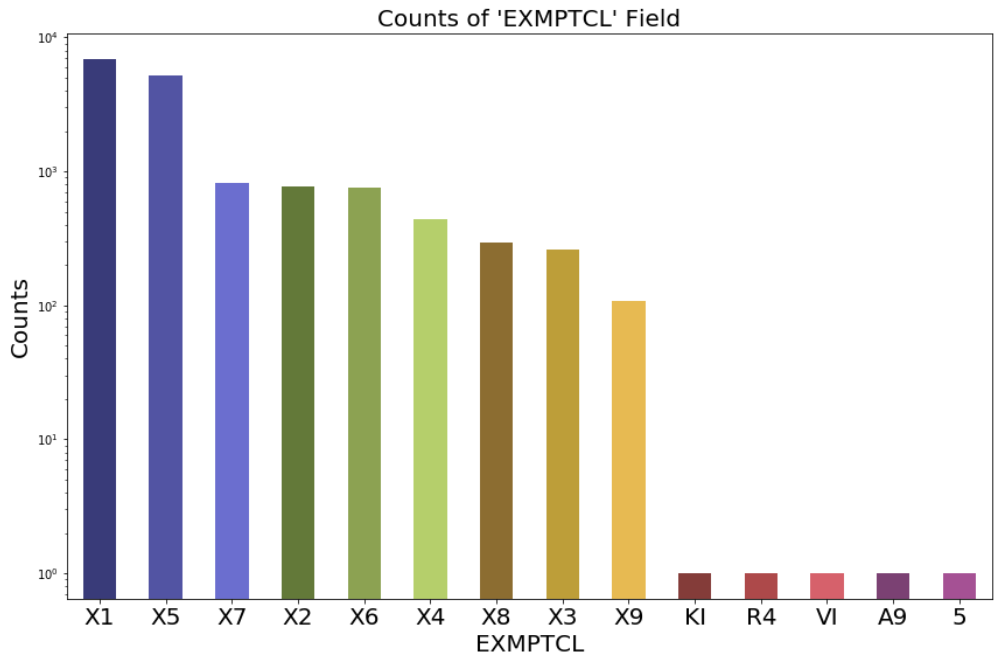
20. STADDR: The STADDR field is the street address for the property.



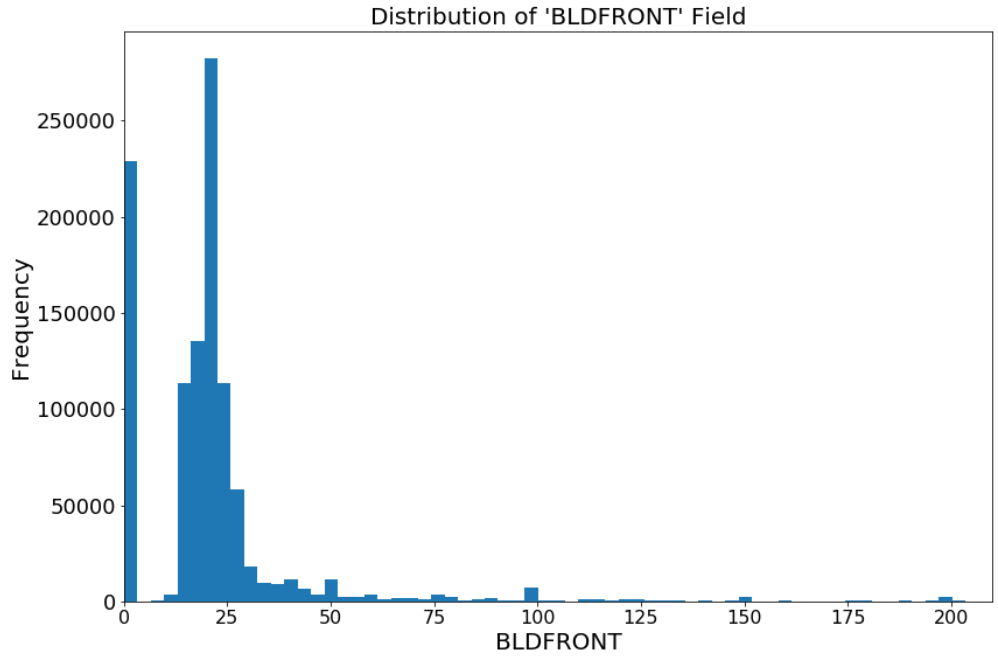
21. ZIP: The ZIP field shows the zip code for the property.



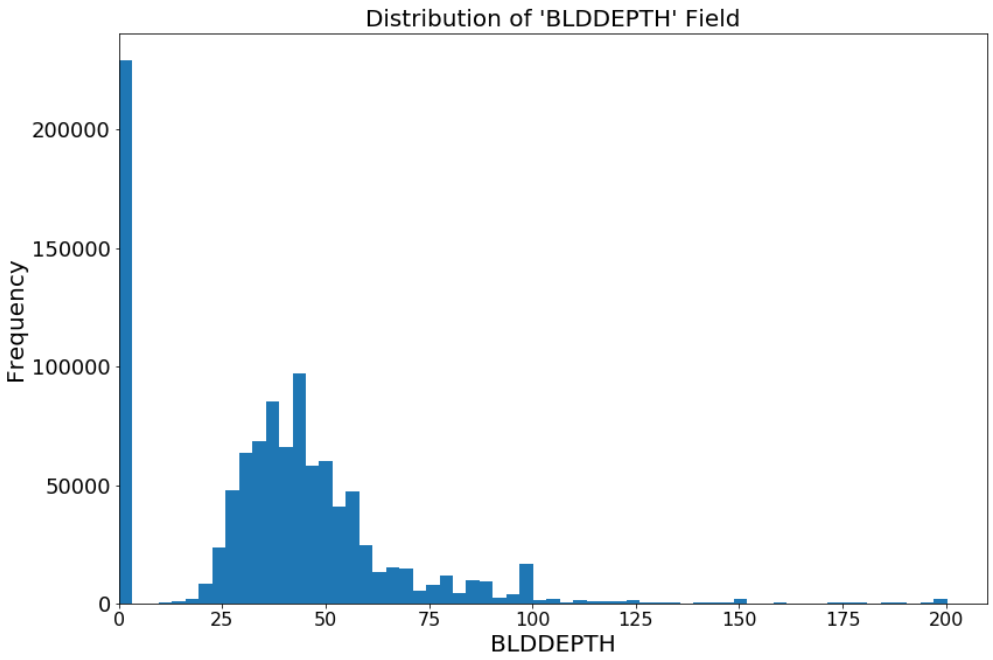
22. EXMPTCL: The EXMPTCL field indicates the exempt class for fully exempt properties.



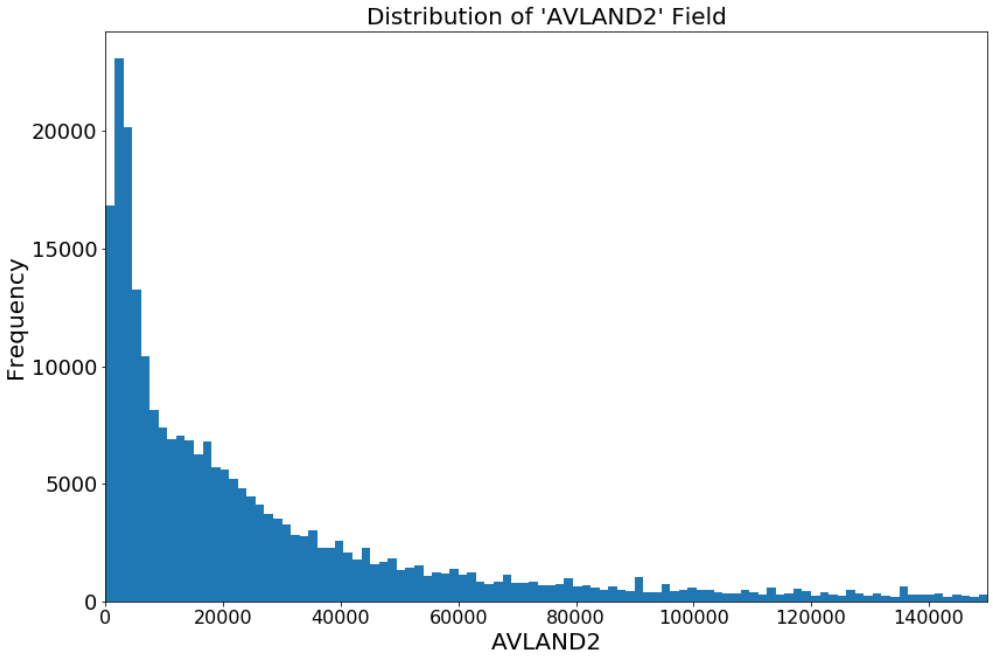
23. BLDFRONT: The BLDFRONT field is the building width in feet. It includes records with value <= 210.



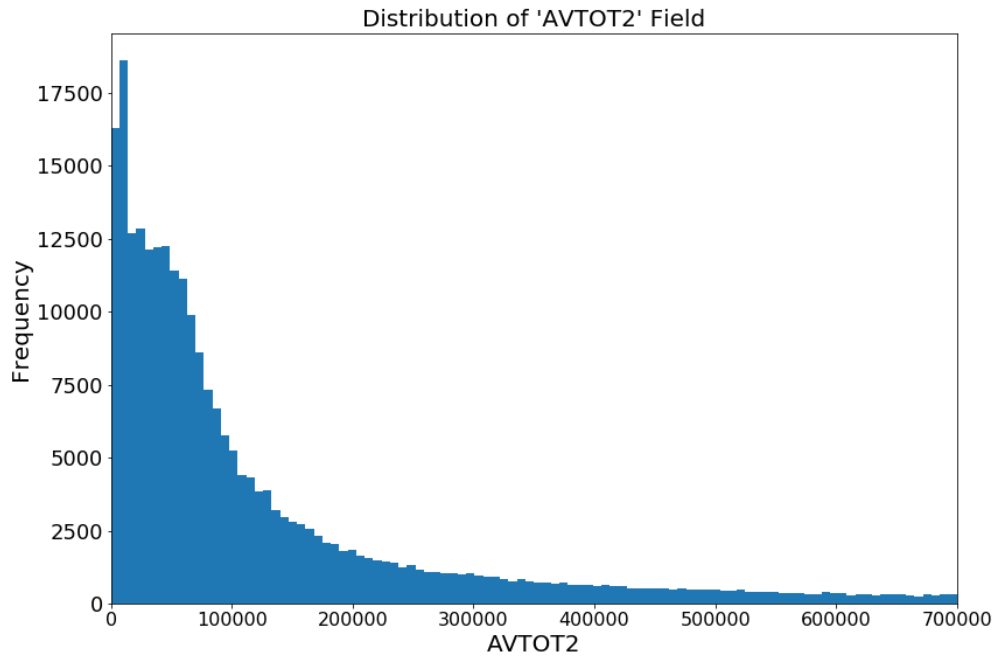
24. BLDDEPTH: The BLDDEPTH field is the building depth in feet. It includes records with value <= 210.



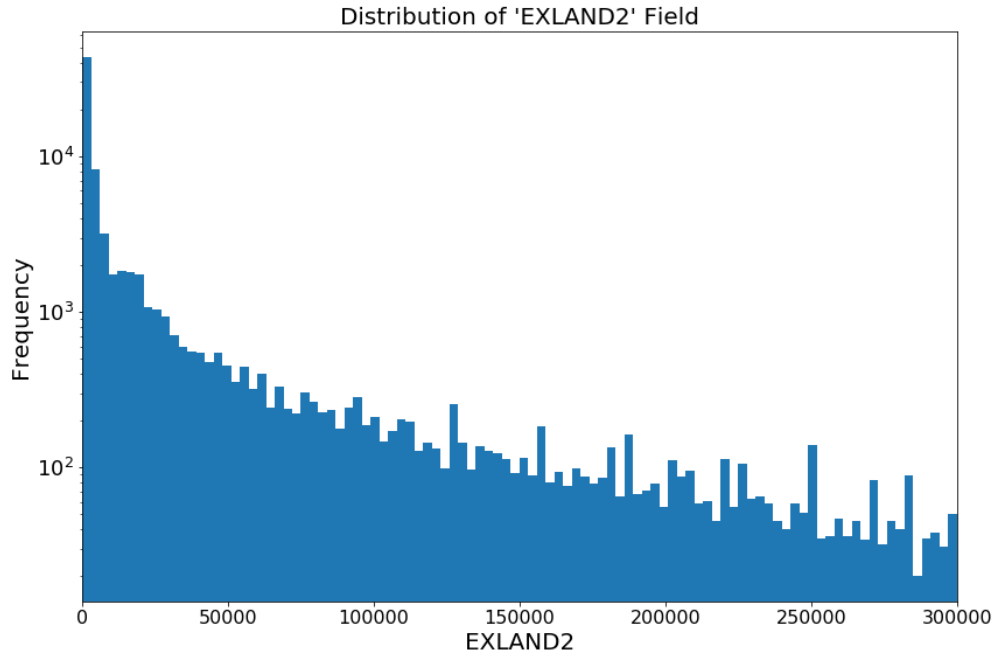
25. AVLAND2: The AVLAND2 field is the transitional land value of the property. It includes records with value <= 150000.



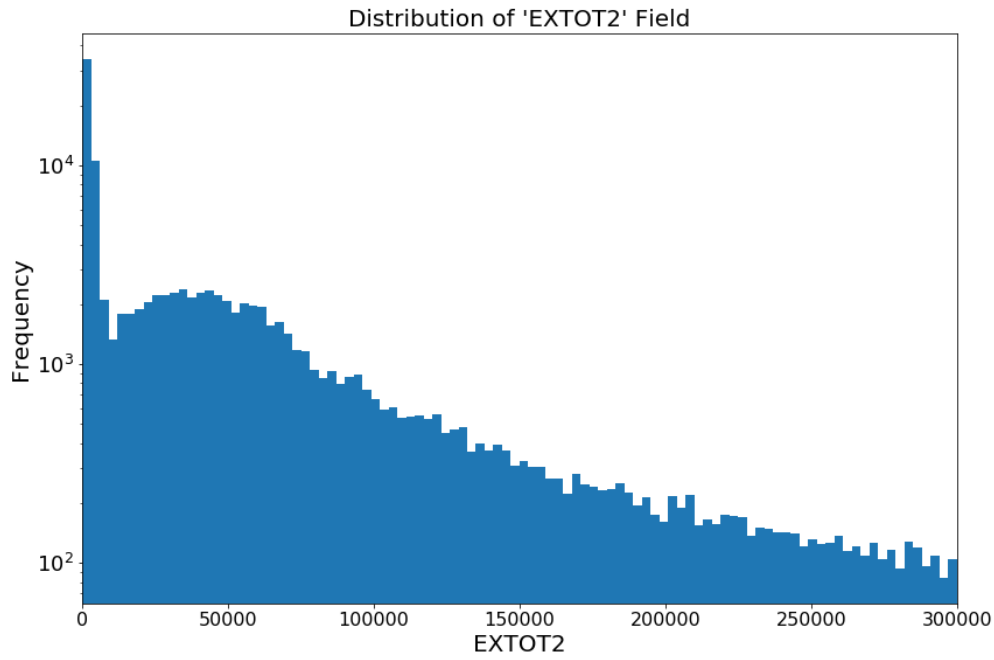
26. AVTOT2: The AVTOT2 field is the transitional total value of the property. It includes records with value <= 700000.



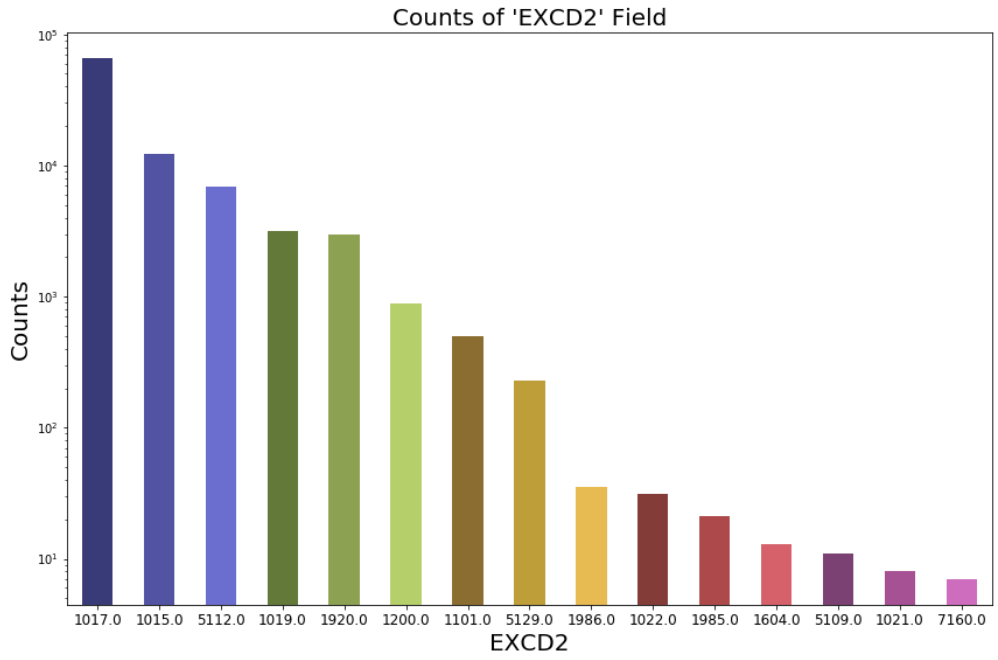
27. EXLAND2: The EXLAND2 field is the transitional exempt land value of the property. It includes records with value <= 300000.



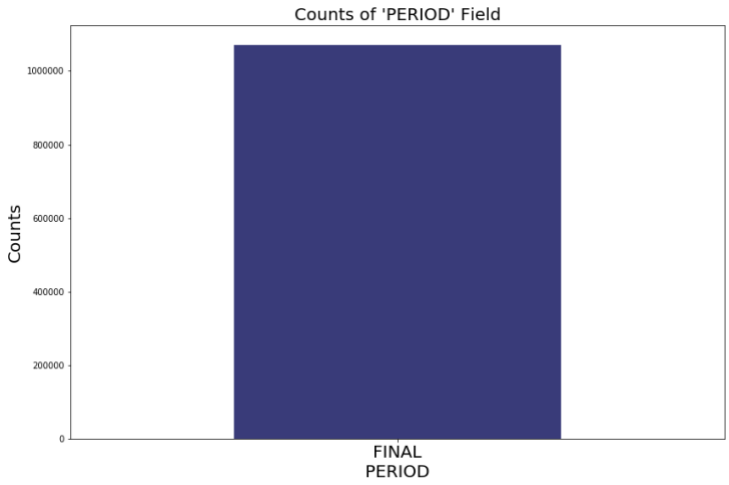
28. EXTOT2: The EXTOT2 field is the transitional exempt total value of the property. It includes records with value <= 300000.



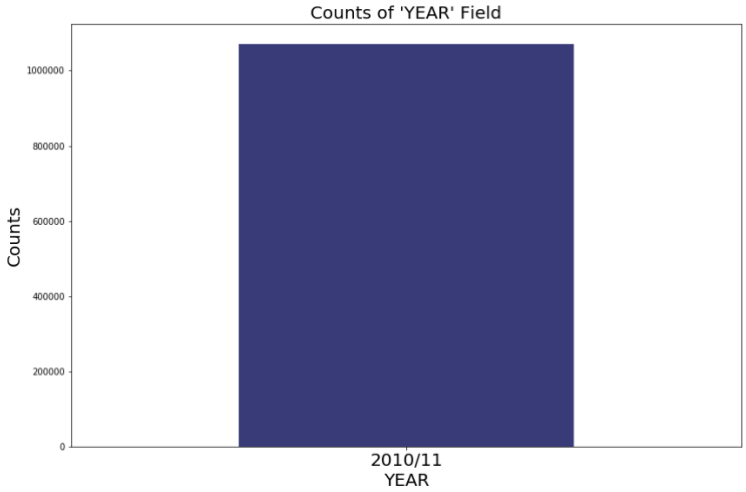
29. EXCD2: The EXCD2 field represents the exemption code 2 for each property.



30. PERIOD: The PERIOD field shows the assessment period when the file was created. The field is filled with 'FINAL' for all records



31. YEAR: The YEAR field indicates the year of assessment, which is filled with ‘2010/11’ for all records.



32. VALTYPE: The VALTYPE field shows the value type, which is filled with ‘AC-TR’ for all records.